UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/658,983	09/10/2003	Dale John Shidla	200310483-1	3966

22879 7590 12/24/2008

HEWLETT PACKARD COMPANY
P O BOX 272400, 3404 E. HARMONY ROAD
INTELLECTUAL PROPERTY ADMINISTRATION
FORT COLLINS, CO 80527-2400

EXAMINER	
DAVE, JYOTI D	
ART UNIT	**PAPER NUMBER**
2191	

NOTIFICATION DATE	DELIVERY MODE
12/24/2008	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

JERRY.SHORMA@HP.COM
mkraft@hp.com
ipa.mail@hp.com

Office Action Summary	Application No.	Applicant(s)	
	10/658,983	SHIDLA ET AL.	
	Examiner	Art Unit	
	JYOTI D. DAVE	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *16 December 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-18* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-18* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *11 September 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *1/2005 and 9/2003*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Claim Rejections - 35 USC § 101

1. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 16-18 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

In claims 16, a “program compiler", "code generator" and "scheduler" are representative of the software. Therefore, claim 6 has been rejected because it is reasonably interpreted as functionally descriptive material, per se. Software, per se, is not one of the statutory subject matter.

Claim 17 is dependent upon claim 1 and are also interpreted as functionally descriptive material.

In claim 18, “computer-readable program”, a "redundant operation" and a “scheduled comparison of results” are representative of the software and/or software functions. Therefore, claim 18 has been rejected because it is reasonably interpreted as functionally descriptive material, per se. Software, per se, is not one of the statutory subject matter.

Claim Rejections - 35 USC § 103

1. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. **Claims 1-3, 5-10, 14-15 and 18 are rejected under 35 U.S.C. 103(a) as being unpatentable over Metzger (7,269,827 B2) in view of Tirumalai (7,234,136) in view of Raina (6,134,675) in further view of Quach (6,640,313 B1).**

In reference to claim 1, Metzger discloses **a method of compiling a program to be executed on a target microprocessor** (see column 1, lines 10-15, in which Metzger discloses Compilers are utilized to convert higher level programming instructions to instructions that may be executed on a particular target computer architecture), **the method comprising scheduling an operation on one of the functional units that would otherwise be idle during a cycle (**see also column 5, lines 60-64, in which Metzger discloses should a functional unit be detected as idle for a period of time exceeding a threshold, then changes may be implemented in the IR, as may be required or useful, to utilize the idle function in a target architecture).

However, Metzger does not disclose **opportunistically** scheduling a **redundant** operation **functional units.** However, Tirumalai discloses **opportunistically**

scheduling a **redundant** operation **with functional units**. (see col. 9, lines 5-15, note that in multiple-issue processor architecture, there are often many unused instructions slots that can be filled with the redundant prefetch operation to potentially avoid a cache miss) Although Tirumalai is disclosing inserting prefretch instructions into code, the concept is similar to the concept disclosed in Metzger (col. 5, lines 60-67, should a functional unit be detected as idle for a period of time exceeding a threshold, then changes may be implemented to the IR, as may be required or useful to utilize the idle function in the target architecture) Tirumalai and Metzger seem to use the same process (scheduling instructions into empty slots) but Tirumalai discloses that those instructions can be redundant instructions.

Metzger also does not disclose **multiple functional units of a same type.** However, Raina discloses **multiple functional units of the same type** (see fig. 1 and col. 1, lines 5-10, core 1, core 2, core 3, and core 4). It would have been obvious to a person skilled in the art at the time of invention to include the comparison method as taught by Raina in order to create an effective testing method. This would have been obvious because Raina clearly teaches that the above process is better suited for creating a faster and improved processor testing method. (Raina, column 1, lines 12-23).

In reference to claim 2, claim 2 is dependent upon claim 1.

Quach disclose **scheduling a comparison** (see column 6, lines 1-7, in which Quach discloses results generated by clusters (a) and (b) [identical instructions to execute clusters (a) and (b)] are compared by check unit and an error is indicated if the execution results are different) **of results from the redundant operation** (see column 7 lines 53-66 and column 8, 1-8, in which Quach discloses For the disclosed embodiment of check unit, each comparator generates a logic value zero when the execution results applied to its inputs match and a logic value one when the execution results don't match). It would have been obvious to a person of ordinary skill in the art at the time of the invention to include the method as taught by Quach in order to create a simple and effective method for indicating when an error in a test comparison has occurred. This would have been obvious because Quach clearly teaches that the above method is better suited for detecting errors in a processing environment. (Quach column 3, lines 22-25)

In reference to claim 3, claim 3 is dependent upon claim 2.

Quach discloses **causing a flag in the target microprocessor to be set when the comparison indicates an error**(see column 7 lines 53-66 and column 8, 1-8, in which Quach discloses For the disclosed embodiment of check unit, each comparator generates a logic value zero when the execution results applied to its inputs match and a logic value one when the execution results don't match). It would have been obvious to a person of ordinary skill in the art at the time of the invention to include the method

as taught by Quach in order to create a simple and effective method for indicating when an error in a test comparison has occurred. This would have been obvious because Quach clearly teaches that the above method is better suited for detecting errors in a processing environment. (Quach column 3, lines 22-25).

In reference to claim 5, Metzger discloses **a method of compiling a program to be executed on a target microprocessor** (see column 1, lines 10-15, in which Metzger discloses Compilers are utilized to convert higher level programming instructions to instructions that may be executed on a particular target computer architecture)**,**

Metzger does not disclose **the method comprising: identifying a cycle during which an operation is available for a first functional unit and no operation is available for a second functional unit.** However, Tirumalai discloses **the method comprising: identifying a cycle during which an operation is available for a first functional unit and no operation is available for a second functional unit** see col. 9, lines 5-15, note that in multiple-issue processor architecture, there are often many unused instructions slots that can be filled with the redundant prefetch operation to potentially avoid a cache miss [inherent that the system must determine if a functional unit is idle and executing the filled functional unit and the idle functional unit]) Although Tirumalai is disclosing inserting prefretch instructions into code, the concept is similar to the concept disclosed in Metzger (col. 5, lines 60-67, should a functional unit be detected as idle for a period of time exceeding a threshold, then changes may be

implemented to the IR, as may be required or useful to utilize the idle function in the target architecture) Tirumalai and Metzger seem to use the same process (scheduling instructions into empty slots) but Tirumalai discloses that those instructions can be redundant instructions. Tirumalai does not specifically disclose **scheduling the operation for execution by both units during the cycle**. However, it would be obvious to one skilled in the art at the time of the invention to include this limitation because Tirumalai discloses filling in instruction slots with prefetch slots. I would be obvious to one skilled in the art that the filled in slot and the originally scheduled slot would process in the same cycle to create a faster performance.

Raina discloses **wherein the first and second function units comprises functional units of a same type** ((see fig. 1 and col. 1, lines 5-10, core 1, core 2, core 3, and core 4). It would have been obvious to a person skilled in the art at the time of invention to include the comparison method as taught by Raina in order to create an effective testing method. This would have been obvious because Raina clearly teaches that the above process is better suited for creating a faster and improved processor testing method. (Raina, column 1, lines 12-23).

Quach discloses **scheduling a comparison of results obtained by the first and second functional units during a subsequent cycle (**see column 6, lines 1-7, in which Quach discloses results generated by clusters (a) and (b) [identical instructions to execute clusters (a) and (b)] are compared by check unit and an error is indicated if the execution results are different)). It is inherent that the check is done after the execution [subsequent cycle] because the system is checking the execution results.

In reference to claim 6, claim 6 is dependent upon claim 5.

Quach discloses **the method of claim 5, wherein the first and second functional units comprise first and second floating point units of the target microprocessor** (see fig. 1, element 158, and column 5, lines 24-30, in which Quach discloses method that tests functional units which comprise of floating point units). It would have been obvious to a person of ordinary skill in the art at the time of invention to include the method as taught by Quach in order to create a simple and effective method for testing processors through comparison. This would have been obvious because Quach clearly teaches that the above method is better suited for detecting errors in a processing environment. (Quach column 3, lines 22-25)

In reference to claim 7, claim 7 is dependent upon claim 5.

Quach discloses **the method of claim 5, wherein the first and second functional units comprise first and second arithmetic logic units of the target microprocessor** (see figure 1 element 154, and column 4, lines 47-48, in which Quach discloses an integer execution unit. Quach further discloses the first and second functional units are of the same type as disclosed above in claim 5. So both functional units can be integer execution units). It would be obvious to one of ordinary skill in the art at the time of the invention to include the method as taught by Quach in order to

create a simple and effective method for testing processors through comparison. This would have been obvious because Quach clearly teaches that the above method I better suited for detecting errors in a processing environment. (Quach column 3, lines 22-25)

In reference to claim 8, claim 8 is dependent upon claim 5.

Quach discloses **the results of the execution are stored in registers within the microprocessor** (Quach discloses a multiple functional units of a same type see column 6, lines 1-2. It would be inherent that the results of the comparison between the results of the execution of the multiple function units are stored in a register within the microprocessor. In all processing systems each functional unit has a register to store the results of the computation). It would be obvious to one of ordinary skill in the art at the time of the invention to include the method as taught by Quach in order to create a simple and effective method for testing processors through comparison. This would have been obvious because Quach clearly teaches that the above method is better suited for detecting errors in a processing environment. (Quach column 3, lines 22-25)

In reference to claim 9, claim 9 is dependent upon claim 5.

Raina discloses **the target microprocessor includes at least three functional units of the same type** (see fig. 1 and col. 1, lines 5-10, core 1, core 2, core 3, and

core 4). It would have been obvious to a person skilled in the art at the time of invention to include the comparison method as taught by Raina in order to create an effective testing method. This would have been obvious because Raina clearly teaches that the above process is better suited for creating a faster and improved processor testing method. (Raina, column 1, lines 12-23).

In reference to claim 10, claim 10 is dependent upon claim 9.

Claim 10 is a compiler claim corresponding to the method of claim 5. Therefore, claim 10 is rejected for the same rational set forth in claim 5

In reference to claim 14, claim 14 is dependent upon claim 5.

Claim 14 is a compiler claim corresponding to the method of claim 3. Therefore, claim 14 is rejected for the same rational set forth in claim 3.

In reference to claim 15, claim 15 is dependent upon claim 14.

Quach discloses **if the error flag is set, then halting the execution and causing a notification to the user of the error flag** (see column 7, line 66 and column 6, line 8, in which Quach discloses there is an error signal created by the comparator when the results do not match). It would have been obvious to a person of ordinary skill

in the art at the time of the invention to include the method as taught by Quach in order to create a simple and effective method for indicating when an error in a test comparison has occurred. This would have been obvious because Quach clearly teaches that the above method is better suited for detecting errors in a processing environment. (Quach column 3, lines 22-25).

In reference to claim 18, Metzger discloses **a computer readable program product for execution on a target microprocessor** (see column 1, lines 10-15, in which Metzger discloses Compilers are utilized to convert higher level programming instructions to instructions that may be executed on a particular target computer architecture), **the program product comprising executable code** (see column 1, lines 10-15, in which Metzger discloses the programming instructions that can be executed on a target computer, the program instructions are executable code) **that includes a operation schedule for one of the functional units that would otherwise be idle (**see also column 5, lines 60-64, in which Metzger discloses should a functional unit be detected as idle for a period of time exceeding a threshold, then changes may be implemented in the IR, as may be required or useful, to utilize the idle function in a target architecture**)** .

However Metzger does not disclose the operation is **redundant.** However, Tirumalai discloses **redundant operations** (see col. 9, lines 5-15, note that in multiple-issue processor architecture, there are often many unused instructions slots that can be filled with the redundant prefetch operation to potentially avoid a cache miss) Although

Tirumalai is disclosing inserting prefretch instructions into code, the concept is similar to the concept disclosed in Metzger (col. 5, lines 60-67, should a functional unit be detected as idle for a period of time exceeding a threshold, then changes may be implemented to the IR, as may be required or useful to utilize the idle function in the target architecture) Tirumalai and Metzger seem to use the same process (scheduling instructions into empty slots) but Tirumalai discloses that those instructions can be redundant instructions.

Metzger does not disclose the program product includes multiple functional units of a same type or a subsequently scheduled comparison of results from the redundant operation for fault checking purposes**.** However, Raina discloses **with multiple functional units of a same type** (see fig. 1, and col. 1, lines 5-10). It would have been obvious to a person skilled in the art at the time of invention to include the comparison method as taught by Raina in order to create an effective testing method. This would have been obvious because Raina clearly teaches that the above process is better suited for creating a faster and improved processor testing method. (Raina, column 1, lines 12-23).

Quach discloses **a subsequent scheduled comparison of results from operation (s**ee column 6, lines 1-7, in which Quach discloses results generated by clusters (a) and (b) [identical instructions to execute clusters (a) and (b)] are compared by check unit and an error is indicated if the execution results are different) **for fault checking purposes** (see column 7 lines 53-66 and column 8, 1-8, in which Quach discloses For the disclosed embodiment of check unit, each comparator generates a

logic value zero when the execution results applied to its inputs match and a logic value one when the execution results don't match). It would have been obvious to a person of ordinary skill in the art at the time of the invention to include the method as taught by Quach in order to create a simple and effective method for indicating when an error in a test comparison has occurred. This would have been obvious because Quach clearly teaches that the above method is better suited for detecting errors in a processing environment. (Quach column 3, lines 22-25).

3. **Claim 4 is rejected under 35 U.S.C. 103(a) as being unpatentable over Metzger (7,269,827 B2) in view of Tirumalai (7,234,136) in view of Raina (6,134,675) in further view of Quach (6,640,313 B1) in further view of Fruehling (6,625,688 B1).**

In reference to claim 4, Fruehling discloses **setting a user selectable level for an aggressiveness of said opportunitisic scheduling** (column 12, lines 60-65, in which Fruehling teaches different levels or modes the PSA can work in. It would have been obvious to a person of ordinary skill in the art at the time of the invention to include the method as taught by Fruehling in order to create more user options for opportunistic scheduling . It is also inherent that the settings can have user selectable levels that are commonly used, allowing users to customize desired settings for any number of computer system operations).

4. **Claims 11, 13 and 16-17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Metzger (7,269,827 B2) in view of Tirumalai (7,234,136) in view of Raina (6,134,675) in further view of Quach (6,640,313 B1) in further view of Chan (5,557,761).**

In reference to claim 11, claim 11 is dependent upon claim 5.

Metzger in view of Quach and in further view of Raina do not disclose the method is performed by a scheduler in a code generator of a program compiler. However, Chan discloses **the method is performed by a scheduler in a code generator of a program compiler** (see column 6, lines 40-45, in which Chan discloses while performing this scheduling function, the code generator (which is included in the software compiler (column 5, lines 52-56)) may move instructions from a subsequent basic block to a prior basic block). It would have been obvious to a person of ordinary skill in the art at the time of the invention to include the method as taught by Chan in order to create the testing and execution of the program faster and more resource-efficient, as taught by Chan. (see Chan, column 6, lines 45-50)

In reference to claim 13, claim 13 is dependent upon claim 11.

Raina discloses **the program compiler comprises a cross compiler run on a different microprocessor** (column 3, lines 7-11, in which Raina discloses the test is

performed by an external testing device). It would have been obvious to one skilled in the art at the time of the invention to include the program compiler comprises cross compiler run on different microprocessors as taught by Raina in order to create an effective testing method. This would have been obvious because Raina clearly teaches the above process is better suited for creating a faster and improved processor testing method. (Raina column 1, lines 12-23)

In reference to claim 16, Metzger discloses **a program compiler for a target microprocessor (**see column 1, lines 10-15, in which Metzger discloses Compilers are utilized to convert higher level programming instructions to instructions that may be executed on a particular target computer architecture**).** Metzger also discloses **a scheduler that schedules a operation on one of the functional units that would otherwise be idle during a cycle (**see also column 5, lines 60-64, in which Metzger discloses should a functional unit be detected as idle for a period of time exceeding a threshold, then changes may be implemented in the IR, as may be required or useful, to utilize the idle function in a target architecture).

Metzger does not disclose with multiple equivalent functional units and the compiler comprising a code generator. However, Quach discloses **with multiple equivalent units** However, Quach discloses with multiple units (see column 6, lines 1-2 in which Quach clearly discloses a method that tests functional units which comprise of logic units of the same type). It would have been obvious to a person of ordinary skill in

the art at the time of the invention to include the method as taught by Quach in order to create a simple and effective method for testing processors through comparison. This would have been obvious because Quach clearly teaches that the above method is better suited for detecting errors in a processing environment,. (Quach column 3, lines 22-25).

Metzger does not disclose **opportunistically** scheduling a **redundant** operation **functional units.** However, Tirumalai discloses **opportunistically** scheduling a **redundant** operation **with functional units**. (see col. 9, lines 5-15, note that in multiple-issue processor architecture, there are often many unused instructions slots that can be filled with the redundant prefetch operation to potentially avoid a cache miss) Although Tirumalai is disclosing inserting prefretch instructions into code, the concept is similar to the concept disclosed in Metzger (col. 5, lines 60-67, should a functional unit be detected as idle for a period of time exceeding a threshold, then changes may be implemented to the IR, as may be required or useful to utilize the idle function in the target architecture) Tirumalai and Metzger seem to use the same process (scheduling instructions into empty slots) but Tirumalai discloses that those instructions can be redundant instructions.

Metzger also does not disclose **multiple functional units of a same type.** However, Raina discloses **multiple functional units of the same type** (see fig. 1 and col. 1, lines 5-10, core 1, core 2, core 3, and core 4). It would have been obvious to a person skilled in the art at the time of invention to include the comparison method as taught by Raina in order to create an effective testing method. This would have been

obvious because Raina clearly teaches that the above process is better suited for creating a faster and improved processor testing method. (Raina, column 1, lines 12-23).

Chan discloses **the compiler comprising a code generator** (see column 5, lines 52-58, in which Chan discloses a software compiler includes a code generator). It would have been obvious to one skilled in the art at the time of the invention to include the method as taught by Chan in order to create a compiler that generates source code. This would have been obvious because Chan clearly teaches that the above method is better suited for a faster executing of the program code and more resource efficient during the execution.

In reference to claim 17, claim 17 is dependent upon claim 16.

Quach disclose **the scheduler subsequently schedules a comparison of results from the redundant operation.** Quach discloses scheduling a comparison of results from the redundant operation (see column 6, lines 1-7, in which Quach discloses results generated by clusters (a) and (b) [identical instructions to execute clusters (a) and (b)] are compared by check unit and an error is indicated if the execution results are different). It would be obvious to one of ordinary skill in the art at the time of the invention to include the method as taught by Quach in order to create a simple and effective method for testing processors through comparison. This would have been

obvious because Quach clearly teaches that the above method is better suited for detecting errors in a processing environment. (Quach column 3, lines 22-25)

5. **Claim 12 is rejected under 35 U.S.C. 103(a) as being unpatentable over Metzger (7,269,827 B2) in view of Tirumalai (7,234,136) in view of Raina (6,134,675) in further view of Quach (6,640,313 B1) in view of Chan (5,557,761) and in further view of Fruehling (6,625,688 B1).**

In reference to claim 12, claim 12 is dependent upon claim 11.

Metzger in view of Quach and in further view of Raina does not disclose the program compiler comprises a native compiler for the target microprocessor. However, Fruehling discloses **the program compiler comprises a native compiler for the target microprocessor** (column 11, lines 58-65, Fruehling discloses the comparator is done within the target device). It would have been obvious to a person of ordinary skill in the art at the time of the invention to include the method as taught by Quach in order to create a simple and effective method for indicating when an error in a test comparison has occurred. This would have been obvious because Quach clearly teaches that the above method is better suited for detecting errors in a processing environment. (Quach column 3, lines 22-25).

Response to Arguments

Applicant's arguments have been fully considered but they are not persuasive for the reasons set forth below:

In reference to claims 1-4:

Previous rejection has been withdrawn. New references have been cited disclosing the limitations of claim 1-4.

Specifically,

First:

New art has been specifically cited disclosing opportunistic scheduling of redundant operation on one of the multiple functional units of the same type and test functional units in the same context as disclosed in the present application.

Second:

Quach has been reexamined and it has been determined Quach is still relevant. Previous arguments have been withdrawn and new arguments have been added.

Examiner contends that Quach is relevant and discloses a commonly used method of comparing data and returning an flag if no match occurred. Although Quach uses this method to detect soft errors, this method could be used to detect other errors as well.

In reference to claim 16-17:

Previous rejections have been withdrawn. New references have been cited disclosing the limitations of claim 16-17.

Specifically,

New art has been specifically cited disclosing opportunistic scheduling of redundant operation on one of the multiple functional units of the same type and test functional units in the same context as disclosed in the present application.

In reference to claim 18:

Previous rejections have been withdrawn. New references have been cited disclosing the limitations of claim 16-17.

Specifically,

New art has been specifically cited disclosing opportunistic scheduling of redundant operation on one of the multiple functional units of the same type and test functional units in the same context as disclosed in the present application.

In reference to claim 5,

Previous rejections have been withdrawn. New references have been cited disclosing the limitations of claim 5.

Specifically,

New art has been specifically cited scheduling the operation for execution by both the first and the second functional units during the cycle and scheduling a comparison of results obtained by the first and the second functional units during a subsequent cycle.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jyoti D. Dave whose telephone number is 571-270-1470. The examiner can normally be reached on 7:30 AM to 5 PM Mon-Fri, Alt Fri. Eastern Time.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Zhen can be reached on 571-272-3708. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a

USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Jyoti D Dave/ 12/16/2008
Examiner, Art Unit 2191
/Wei Y Zhen/

Supervisory Patent Examiner, Art Unit 2191